TIssueV2 - Bug # 1906: invalid json return for API

Status:	Closed	**Priority:**	Normal
Author:	Lauri Carpenter	**Category:**	
Created:	09/23/2011	**Assigned to:**	Lauri Carpenter
Updated:	10/25/2011	**Due date:**	
Subject:	invalid json return for API		

Description:

As reported by Tim Rupp.

<pre>

The TIssue API does not return valid JSON

It returns a string that has a preamble word, such as "OK" or "FAIL",
and then a string of text which is JSON.

As a result of this return value, one cannot use standard JSON libraries
to directly parse the responses from TIssue.

Instead, one needs to massage the data, transforming it into valid JSON,
and then run it through the appropriate decoder.

An example of the body of the HTTP response from TIssue is shown below.

[body:protected] =>
OK
{
  "event_id": 992,
  "event_state": "O",
  "event_behavior_description": "Normal event creation",
  "event_created": true
}

As you can see, the word "OK" resides outside of the JSON code (which
begins at the first brace)

In PHP, json_decode returns NULL when trying to decode this. Note that
invalid JSON, using the PHP libraries, always returns NULL.

 http://us3.php.net/manual/en/function.json-decode.php

In Perl, the JSON module throws the error
"""
malformed JSON string, neither array, object, number, string or atom, at
character offset 0 (before "OK\n{\n   "event_id...")
"""

In Python, decoding the response raises an exception

[root@cst-dev-tst tissue]# python2.7 ok.py
Traceback (most recent call last):
 File "ok.py", line 15, in <module>
  print json.loads(myvar)

```
 File "/usr/local/lib/python2.7/json/__init__.py", line 326, in loads
   return _default_decoder.decode(s)
 File "/usr/local/lib/python2.7/json/decoder.py", line 366, in decode
   obj, end = self.raw_decode(s, idx=_w(s, 0).end())
 File "/usr/local/lib/python2.7/json/decoder.py", line 384, in raw_decode
   raise ValueError("No JSON object could be decoded")
ValueError: No JSON object could be decoded
```

If I remove the "OK" text from the response, it decodes as valid JSON in the libraries that I have tried. They are,

PHP: json_decode function
Perl: JSON module from CPAN
Python: json module from Python 2.7+

```
[root@cst-dev-tst tissue]# python2.7 ok.py
{u'event_id': 992, u'event_state': u'O', u'event_created': True,
u'event_behavior_description': u'Normal event creation'}
```

```
[root@klogger ~]# perl js.pl
$VAR1 = {
        'event_created' => bless( do{\(my $o = 1)},
'JSON::backportPP::Boolean' ),
        'event_id' => 992,
        'event_behavior_description' => 'Normal event creation',
        'event_state' => 'O'
      };
```

```
[root@cst-dev-tst tissue]# php sandbox.php
stdClass Object
(
   [event_id] => 992
   [event_state] => O
   [event_behavior_description] => Normal event creation
   [event_created] => 1
)
```

This presents a problem to the development of client side tools.

It falls back on the client developers to develop special handling of the HTTP responses instead of being able to rely on standard JSON parsing libraries available in the particular language

If a status code for the HTTP response is desired, it could be encoded as part of the JSON string itself; such as

[body:protected] =>
{
  "status": "OK",
  "message": {
    "event_id": 992,
    "event_state": "O",
    "event_behavior_description": "Normal event creation",
    "event_created": true
  }
}

Or some other such name for the status of the HTTP response. This has several benefits

1. It would accomplish the need of specifying the appropriate response
2. It is valid JSON that can be parsed with standard libraries
3. It allows for future expansion of the value of the status without needing to change clients; for instance if the need arises to add EXISTS, NOT FOUND, or TOO OLD (for instance if specifying a very old detected_at_time) to OK and FAIL
4. It separates the status of the response from the content of the response. This appears to be the initial goal of the status preamble.

The invalid JSON is the most significant problem at this time.

Thanks,
Tim
</pre>

History

09/29/2011 08:05 am - Lauri Carpenter

- Status changed from New to Resolved

- Assigned to set to Lauri Carpenter

Moved to common ServiceProxy from ncis_common/NcisUtility which parses the json for tissue_api and fbi_api; the showPageApi method in tissue_gui/fbi_gui has been modified to write a dict containing status and result objects; the tissue_exception/fbi_exception middleware.py has been modified to set the result object to the dict of errcls, errmsg and argdict. tarupp has tested against development and is happy. Will be in next release.

10/25/2011 02:09 pm - Randy Reitz

- Status changed from Resolved to Closed